Learnable pooling with Context Gating for video classification

Antoine Miech[1,2] Ivan Laptev [1,2] Josef Sivic[1,2,3]
[1]École Normale Supérieure* [2]Inria [3]CIIRC†

https://github.com/antoine77340/LOUPE

Abstract

Common video representations often deploy an average or maximum pooling of pre-extracted frame features over time. Such an approach provides a simple means to encode feature distributions, but is likely to be suboptimal. As an alternative, we here explore combinations of learnable pooling techniques such as Soft Bag-of-words, Fisher Vectors, NetVLAD, GRU and LSTM to aggregate video features over time. We also introduce a learnable non-linear network unit, named Context Gating, aiming at modeling interdependencies between features. We evaluate the method on the multi-modal Youtube-8M Large-Scale Video Understanding dataset using pre-extracted visual and audio features. We demonstrate improvements provided by the Context Gating as well as by the combination of learnable pooling methods. We finally show how this leads to the best performance, out of more than 600 teams, in the Kaggle Youtube-8M Large-Scale Video Understanding challenge.

1. Introduction

Understanding and recognizing video content is one of the major challenges in computer vision. Applications include surveillance, personal assistance, smart homes, autonomous driving, stock footage search and sports video analysis. Current methods for video analysis typically represent videos by features extracted from one or several consecutive frames, followed by feature aggregation over time. Example methods for feature extraction include deep convolutional neural network (CNN) features pre-trained on static images [15, 25, 36, 39] or short video clips [40, 12], as well as hand-crafted video features [26, 34, 42]. Common methods for feature aggregation include simple temporal averaging or max-pooling as well as more sophisticated pooling techniques such as VLAD [20] as well as temporal models such as LSTM [17] and GRU [7].

Figure 1: Two example videos from the Youtube-8M V2 dataset together with the ground truth and predicted labels. Predictions are color-coded as (green): correct, (orange): correct but missing in the ground truth, (red) incorrect.

In this work we make the following two contributions. First we explore temporal aggregation of visual and audio features by designing new and comparing existing learnable pooling methods such as NetVLAD [3], GRU [7] and LSTM [17]. We investigate each method individually and demonstrate their complementarity through combination. Second, inspired by [9], we introduce a non-linear learnable network unit, named Context Gating (CG). CG aims at better capturing the non-linear interdependencies between features as well as among output labels.

We evaluate our method on the multi-modal Youtube-8M V2 dataset containing about 8M videos and 4716 unique tags. We use pre-extracted visual and audio features provided with the dataset [2] and demonstrate improvements obtained with the Context Gating as well as by the combination of learnable poolings. Our method obtains best performance, out of more than 600 teams, in the Kaggle Youtube-8M Large-Scale Video Understanding challenge. Figure 1 illustrates some qualitative results of the method.

*[1]Département d'informatique de l'ENS, École normale supérieure, CNRS, PSL Research University, 75005 Paris, France.
†[3]Czech Institute of Informatics, Robotics and Cybernetics at the Czech Technical University in Prague.

2. Related work

This work is related to previous methods on video features extraction and feature aggregation reviewed below.

Feature extraction. Successful hand-crafted representations [26, 34, 42] are based on local histograms of image and motion gradient orientations extracted along dense trajectories [10, 42]. More recent methods extract deep convolutional neural network activations computed from individual frames or blocks of frames using spatial [12, 23, 14, 43] or spatio-temporal [4, 6, 21, 40, 41] convolutions. Convolutional neural networks can be also applied separately on the appearance channel and the pre-computed motion field channel resulting in the, so called, two-stream representations [6, 12, 14, 35, 41].

Feature aggregation. Video features are typically extracted from individual frames or short video clips. The remaining question is: how to aggregate video features over the entire and potentially long videos? One way to achieve this is to employ recurrent neural networks, such as long short-term memory (LSTM) [17] or gated recurrent unit (GRU) [7]), on top of the extracted frame-level features to capture the temporal structure of video into a single representation [5, 11, 18, 27, 46]. Other methods capture only the *distribution* of features in the video, not explicitly modeling their temporal ordering. The simplest form of this approach is the average or maximum pooling of video features [44] over time. Other commonly used methods include bag-of-visual-words [8, 37], Vector of Locally aggregated Descriptors (VLAD) [20] or Fisher Vector [31] encoding. Application of these techniques to video include [26, 29, 34, 42, 45]. The variants of these methods [27, 32] rely on an unsupervised learning of the codebook. However, the codebook can be also learnt in a discriminative manner [29, 30, 38] or the entire encoding module can be included within the convolutional neural network architecture and trained in an end-to-end manner [3]. This type of end-to-end trainable orderless aggregation has been recently applied to video in [14].

3. Multi-label video classification architecture

Overall architecture. Our architecture for video classification is illustrated in Figure 2 and contains three main modules. First, the input features are extracted from the video. Next, the pooling module aggregates the extracted features into a single compact (e.g. 1024-dimensional) representation for the entire video. Finally, the classification module takes the resulting video representation as input and outputs a set of labels for the video together with their scores. The three modules are described next.

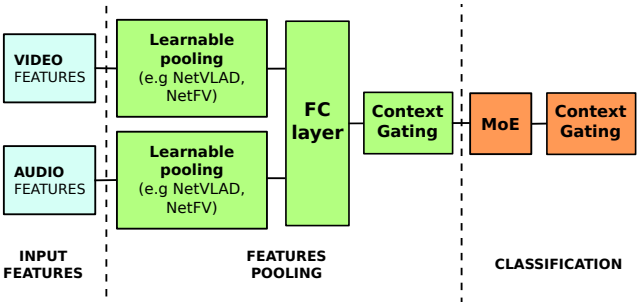

Figure 2: Overview of our network architecture for video classification (the "Late Concat" variant). FC denotes a Fully-Connected layer. MoE denotes the Mixture-of-Experts classifier [2])

Feature extraction. In the Youtube 8M competition [2] video and audio features are provided for every second of the input video. The visual features consist of ReLu activations of the last fully-connected layer from a publicly available Inception network[1] trained on Imagenet. The audio features are extracted from a CNN architecture trained for audio classification [16]. PCA and whitening are then applied to reduce the dimension to 1024 for the visual features and 128 for the audio features.

Feature pooling. The pooling module has a two-stream architecture that takes the visual and audio features as input. Each modality is then processed separately by a learnable pooling method (Section 4) into a single representation. These individually pooled representations are then concatenated and fed into a fully-connected layer to reduce their dimension into a compact (1024-dimensional) vector. This compact vector is then reweighed by the Context Gating layer (Section 5) capturing non-linear interdependencies among features.

Classification. The classification module is composed of a soft Mixture-of-Experts (MoE) classifier [22] as described in [2]. This is followed by the Context Gating layer that reweights the output class probabilities according to learnt prior structure of the output label space (Section 5).

4. Learnable pooling methods

Within our video classification architecture described above, we investigate several types of learnable pooling models, which we describe next.

Pooling via clustering. We explore end-to-end trainable variants of the following three pooling techniques: Bag-of-visual-words [8, 37], VLAD [20] and Fisher Vector [31].

[1] https://www.tensorflow.org/tutorials/image_recognition

For VLAD encoding, we use the NetVLAD [3] architecture proposed for place recognition [3] and then extended to action recognition in video [14]. As opposed to the original version of NetVLAD [3], we did not pre-train the codebook with a k-means initialization as we did not notice any improvement by doing so. We have also investigated a modification of the original NetVLAD architecture that averages the actual descriptors instead of the residuals. We call this variant NetRVLAD (for Residual-less VLAD). This is a simplification of NetVLAD that uses less parameters and needs less computing operations (about half in both cases).

For bag-of-visual-words encoding, we use soft-assignment of descriptors to visual word clusters [3, 33] to obtain a differentiable representation. We call this representation Soft-DBoW (for Soft Deep Bag-of-visual-Words).

Finally, for Fisher Vector encoding, we modify the NetVLAD architecture to allow learning of second order feature statistics within the clusters. We will denote this as NetFV (for Net Fisher Vector) as it is an end-to-end trainable variant of the Fisher Vector [31].

Recurrent models for pooling. We have investigated Long Short-Term Memory (LSTM) [17], and Gated Recurrent Units (GRU) [7]. For both of these models, we stacked two networks with hidden layers of size 1024. The pooled representation is the final state of the network after processing the whole sequence of features in the video.

5. Context Gating

In this section we describe the Context Gating (CG) layer that transforms the input feature representation X into a new representation Y. The layer has the following form:

$$Y = \sigma(WX + b) \circ X, \tag{1}$$

where $X \in \mathbb{R}^n$ is vector of the input feature activations, σ is the element-wise sigmoid activation and $\circ$ is the element-wise multiplication. $W \in \mathbb{R}^{n \times n}$ and $b \in \mathbb{R}^n$ are trainable parameters. The vector of weights $\sigma(WX + b)$ acts as a set of learnt gates (with values between 0 and 1) on the individual dimensions of the input feature X.

The motivation behind this transformation is two-fold. First, we wish to introduce non-linear interactions among activations of the input representation. Second, we wish to recalibrate the strengths of different activations of the input representation through a self-gating mechanism. The form of the Context Gating layer is inspired by the Gated Linear Unit (GLU) introduced recently for language modeling [9] that considers a more complex class of transformations given by $\sigma(W_1X + b_1) \circ (W_2X + b_2)$, with two sets of learnable parameters W_1, b_1 and W_2, b_2. Compared to the the Gated Linear Unit [9], our Context Gating in (1) (i) reduces the number of learnt parameters as only one set of weights is learnt, and (ii) re-weights directly the input vector X (instead of its linear transformation) and hence is suitable for situations where X has a specific meaning, such the score of a class label, that is preserved by the layer. As shown in Figure 2, we use Context Gating in the feature pooling and classification modules. First, we use CG to transform the feature vector before passing it to the classification module. Second, we use CG after the classification layer to capture the prior structure of the output label space. Details are provided below.

Capturing dependencies among features. First, we place the Context Gating layer to transform the feature vector just before the classifier. The aim is to capture the dependencies among the features. For example, the context gating can learn to suppress features likely to be on background and emphasize the foreground objects. For instance, if features corresponding to 'Trees', 'Skier' and 'Snow' have high co-occurring activations in a skiing video, context gating could learn to suppress the background features such as 'Trees' and 'Snow', which are less important for the classification.

Capturing prior structure of output space. We also place the Context Gating layer after the Mixture of Experts classifier to re-weight the output probabilities for the different classes. For instance, the output label pair 'Make Up' and 'Car' is much less likely than the label pair 'Renault' and 'Car'. Context gating aims at downweighting such unlikely label combinations in the output.

6. Training details

All models are trained using the Adam algorithm [24] and mini-batches with around 100 frames. The learning rate is initially set to 0.0002 and is then decreased exponentially with the factor of 0.8 every 4M samples. We use gradient clipping and batch normalization [19] before each non-linear layer.

For the clustering-based pooling models, i.e. Soft-DBoW, NetVLAD, NetRVLAD and NetFV, we randomly sample N features with replacement from each video. N is fixed for all videos at training and testing. For training of recurrent models, i.e. LSTM and GRU, we process features in the temporal order. We have also experimented with the random sampling of frames for LSTM and GRU which performs surprisingly similarly.

All our models are trained with the cross entropy loss. We found this loss to work well for maximizing the Global Average Precision (GAP) metric. Our implementation uses the TensorFlow framework [1]. Each training is performed on a single NVIDIA TITAN X (12Gb) GPU.

Method	GAP
Baseline 1 (Average pooling + Logistic Regression)	71.4%
Baseline 2 (Average pooling + MoE + CG)	74.1%
LSTM (2 Layers)	81.7%
GRU (2 Layers)	82.0%
Soft-DBoW (4096 Clusters)	81.6%
NetFV (128 Clusters)	82.2%
NetVLAD (256 Clusters)	82.4%
Gated Soft-DBoW (4096 Clusters)	82.0%
Gated NetFV (128 Clusters)	83.0%
Gated NetRVLAD (256 Clusters)	83.1%
Gated NetVLAD (256 Clusters)	**83.2%**

Table 1: Performance comparison for individual aggregation schemes. Clustering-based methods are compared with and without Context Gating.

Method	GAP
NetVLAD	82.2%
NetVLAD + CG after pooling	82.7%
NetVLAD + GLU after pooling, CG after MoE	82.7%
NetVLAD + CG after pooling and MoE	**83.0%**

Table 2: Evaluation of Context Gating for the NetVLAD-based architecture with 128 clusters.

Method	Early Concat	Late Concat
NetVLAD	81.9%	**82.4%**
NetFV	81.2%	**82.2%**
GRU	**82.2%**	82.1%
LSTM	**81.7%**	81.1%

Table 3: Evaluation of audio-video fusion methods (Early and Late Concat).

7. Experiments

Youtube-8M Dataset. The Youtube-8M dataset [2] is composed of approximately 8 millions videos. Visual and audio features are pre-extracted and provided with the dataset for each second of the video. Visual features are obtained by the state-of-the-art Inception CNN followed by the PCA-compression into a 1024 dimensional vector. More details on feature extraction are available in [2].

Each video is labeled with one or multiple tags referring to the main topic of the video. For instance, a video showing someone making a chocolate cake may have labels 'Food', 'Cooking', 'Cake' and 'Chocolate' from the full set of 4716 tags. Two example videos and corresponding annotations from Youtube-8M are illustrated in Figure 1.

The original dataset is divided into training, validation and test subsets with 70%, 20% and 10% of videos, respectively. In this work we keep around 20K videos for the validation, the remaining samples from the original training and validation subsets are used for training. This choice was made to obtain a larger training set and to decrease the validation time. We have noticed that the performance on our validation set was comparable (0.2%-0.3% higher) to the test performance evaluated on the Kaggle platform. As we have no access to the test labels, all results in this section are reported for our validation set. We report evaluation using the Global Average Precision (GAP) metric at top 20 as used in the Youtube-8M Kaggle competition.

Model evaluation. We evaluate the performance of individual models in Table 1. To enable a fair comparison, all pooled representations have the same size of 1024 dimensions. The "Gated" versions for the clustering-based pooling methods include CG layers as described in Section 5. Using CG layers together with GRU and LSTM has decreased performance in our experiments.

From Table 1 we can observe a significant increase of performance provided by all learnt aggregation schemes compared to the Average pooling baselines. Interestingly, the NetVLAD and NetFV representations based on the temporally-disordered feature pooling outperforms the temporal models (GRU and LSTM). Finally, we can note a consistent increase in performance provided by the CG for all clustering-based pooling methods.

Context Gating. Table 2 presents an ablation study evaluating the effect of Context Gating on the NetVLAD aggregation with 128 clusters. The addition of CG layers in the feature pooling and classification modules gives a significant increase in GAP. We have observed a similar behavior for NetVLAD with 256 clusters. We also experimented with replacing the Context Gating by the GLU [9] after pooling. To make the comparison fair, we added a Context Gating layer just after the MoE. Despite being less complex than GLU, we observe that CG also performs better. We note that the improvement of 0.8% provided by CG is similar to the improvement of the best non-gated model (NetVLAD) over LSTM in Table 1.

Video-Audio fusion. In addition to the late fusion of audio and video streams (Late Concat) described in Section 3, we have also experimented with a simple concatenation of original audio and video features into a single vector, followed by the pooling and classification modules in a "single stream manner" (Early Concat). Results in Table 3 illustrate the effect of the two fusion schemes for different

pooling methods. The two-stream audio-visual architecture with the late fusion results in improved performance for the clustering-based pooling methods (NetVLAD and NetFV). On the other hand, the early fusion scheme seems to work better for GRU and LSTM aggregations. We have also experimented with replacing the concatenation fusion of audio-video features by their outer product. We found this did not work as well as concatenation mainly due to the high dimensionality of the resulting output. To alleviate this issue, we tried to reduce the output dimension using the multi-modal compact bilinear pooling approach [13] but found the resulting models underfitting the data.

8. Ensembling

In this section we explore the complementarity of different models and consider their combination through ensembling. Our ensemble consists of several independently trained models. The ensembling averages label prediction scores of selected models. We have observed the increased effect of ensembling when combining diverse models. The ensemble did not bring much when combining best but similar models. To choose models, we follow a simple greedy approach: we start with the best performing model and choose the next model by maximizing the GAP of the ensemble on the validation set. Our final ensemble used in the Youtube 8M challenge contains 25 models. From results in Figure 3 we observe that most of the improvements are obtained by ensembling the first seven models. A seven models ensemble is enough to reach the first place with a GAP on the private test set of 84.698. These seven models correspond to: Gated NetVLAD (256 clusters), Gated NetFV (128 clusters), Gated Soft-DBoW (4096 Clusters), Soft-DBoW (8000 Clusters), Gated NetRVLAD (256 Clusters), GRU (2 layers, hidden size: 1200) and LSTM (2 layers, hidden size: 1024). Our code to reproduce this ensemble is available from https://github.com/antoine77340/Youtube-8M-WILLOW. To obtain more diverse models for the final 25 ensemble, we also added all the non-Gated models, varied the number of clusters or varied the size of the pooled representation.

9. Conclusions

We have addressed the problem of large-scale video tagging and explored trainable variants of classical pooling methods (BoW, VLAD, FV) for the temporal aggregation of audio and visual features. In this context we have observed NetVLAD, NetFV and Soft-DBoW to outperform more common temporal models such as LSTM and GRU. We have also introduced the Context Gating mechanism and have shown its benefit for the trainable versions of BoW, VLAD and FV. The ensemble of our individual models have been shown to improve the performance further, enabling

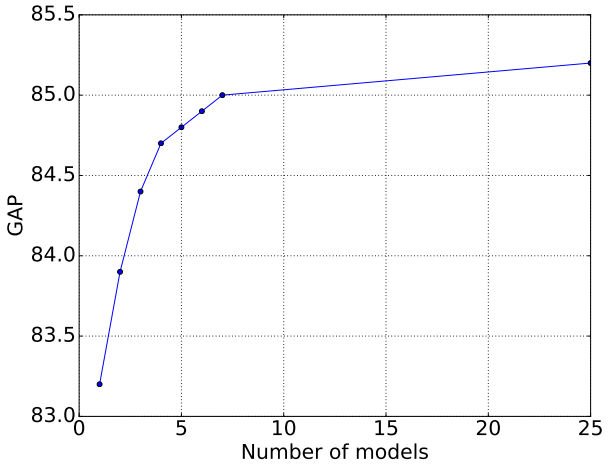

Figure 3: The performance (GAP) of video tagging in our validation set over the number of models combined in the ensemble. The GAP scores for the public and private test sets of the Youtube 8M challenge are approximatively 0.2% lower than on our validation set.

our method to win the Youtube 8M Large-Scale Video Understanding Kaggle challenge. Our TensorFlow toolbox LOUPE is available for download from [28] and includes implementations of the Context Gating as well as learnable pooling modules used in our work.

Acknowledgments

The authors would like to thanks Jean-Baptiste Alayrac and Relja Arandjelović for valuable discussions. Google for providing their Youtube-8M Tensorflow Starter Code. This work has also been partly supported by ERC grants ACTIVIA (no. 307574) and LEAP (no. 336845), CIFAR Learning in Machines & Brains program, ESIF, OP Research, development and education Project IMPACT No. CZ.02.1.01/0.0/0.0/15_003/0000468 and a Google Research Award.

References

[1] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mane, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viegas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467*, 2015. 3

[2] S. Abu-El-Haija, N. Kothari, J. Lee, P. Natsev, G. Toderici, B. Varadarajan, and S. Vijayanarasimhan. Youtube-8m: A large-scale video classification benchmark. *arXiv preprint arXiv:1609.08675*, 2016. 1, 2, 4

[3] R. Arandjelovic, P. Gronat, A. Torii, T. Pajdla, and J. Sivic. NetVLAD: CNN architecture for weakly supervised place recognition. In *CVPR*, 2016. 1, 2, 3

[4] M. Baccouche, F. Mamalet, C. Wolf, C. Garcia, and A. Baskurt. Sequential deep learning for human action recognition. *Human Behavior Understanding*, pages 29–39, 2011. 2

[5] F. Basura, E. Gavves, J. M. Oramas, A. Ghodrati, and T. Tuytelaars. Modeling video evolution for action recognition. In *CVPR*, 2015. 2

[6] J. Carreira and A. Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *CVPR*, 2017. 2

[7] K. Cho, B. van Merrienboer, D. Bahdanau, and Y. Bengio. On the Properties of Neural Machine Translation: Encoder-Decoder Approaches. *arXiv preprint arXiv:1409.1259*, 2014. 1, 2, 3

[8] G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray. Visual categorization with bags of keypoints. In *ECCV Workshop*, 2004. 2

[9] Y. N. Dauphin, F. Angela, M. Auli, and D. Grangier. Language modeling with gated convolutional networks. In *arXiv preprint arXiv:1612.08083*, 2016. 1, 3, 4

[10] C. R. de Souza, A. Gaidon, E. Vig, and A. M. López. Sympathy for the details: Dense trajectories and hybrid classification architectures for action recognition. In *ECCV*, 2016. 2

[11] J. Donahue, L. A. Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell. Long-term recurrent convolutional networks for visual recognition and description. *arXiv preprint arXiv:1411.4389*, 2014. 2

[12] C. Feichtenhofer, A. Pinz, and A. Zisserman. Convolutional two-stream network fusion for video action recognition. In *CVPR*, 2016. 1, 2

[13] Y. Gao, O. Beijbom, N. Zhang, and T. Darrell. Compact bilinear pooling. In *CVPR*, 2016. 5

[14] R. Girdhar, D. Ramanan, A. Gupta, J. Sivic, and B. Russell. Actionvlad: Learning spatio-temporal aggregation for action classification. In *CVPR*, 2017. 2, 3

[15] K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. In *CVPR*, 2016. 1

[16] S. Hershey, S. Chaudhuri, D. P. W. Ellis, J. F. Gemmeke, A. Jansen, C. Moore, M. Plakal, D. Platt, R. A. Saurous, B. Seybold, M. Slaney, R. Weiss, and K. Wilson. CNN architectures for large-scale audio classification. In *International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017. 2

[17] S. Hochreiter and J. Schmidhuber. Long short-term memory. In *Neural Computing*, 1997. 1, 2, 3

[18] M. Ibrahim, S. Muralidharan, Z. Deng, A. Vahdat, and M. Greg. A Hierarchical Deep Temporal Model for Group Activity Recognition. In *CVPR*, 2016. 2

[19] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate. *arXiv preprint arXiv:1502.03167*, 2015. 3

[20] H. Jegou, M. Douze, C. Schmid, and P. Perez. Aggregating local descriptors into a compact image representation. In *CVPR*, 2010. 1, 2

[21] S. Ji, W. Xu, M. Yang, and K. Yu. 3D Convolutional Neural Networks for Human Action Recognition. In *PAMI*, 2013. 2

[22] M. I. Jordan. Hierarchical mixtures of experts and the em algorithm. *Neural Computation*, 1994. 2

[23] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei. Large-scale video classification with convolutional neural networks. In *CVPR*, pages 1725–1732, 2014. 2

[24] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 3

[25] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 1

[26] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. Learning realistic human actions from movies. In *CVPR*, 2008. 1, 2

[27] G. Lev, G. Sadeh, B. Klein, and L. Wolf. Rnn fisher vectors for action recognition and image annotation. In *ECCV*, 2016. 2

[28] A. Miech. LOUPE tensorflow toolbox for learnable pooling module. https://github.com/antoine77340/LOUPE, 2017. 5

[29] X. Peng, L. Wang, Y. Qiao, and Q. Peng. Boosting VLAD with Supervised Dictionary Learning and High-Order Statistics. In *ECCV*, 2014. 2

[30] X. Peng, C. Zou, Y. Qiao, and Q. Peng. Action recognition with stacked fisher vectors. In *ECCV*, 2014. 2

[31] F. Perronnin and C. Dance. Fisher kernels on visual vocabularies for image categorization. In *CVPR*, 2007. 2, 3

[32] F. Perronnin and D. Larlus. Fisher Vectors Meet Neural Networks: A Hybrid Classification Architecture. In *CVPR*, 2015. 2

[33] J. Philbin, O. Chum, M. Isard, J. Sivic, and A. Zisserman. Lost in quantization: Improving particular object retrieval in large scale image databases. In *CVPR*, 2008. 3

[34] C. Schüldt, I. Laptev, and B. Caputo. Recognizing human actions: a local svm approach. In *ICPR*, 2004. 1, 2

[35] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *ICLR*, pages 568–576, 2014. 2

[36] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 1

[37] J. Sivic and A. Zisserman. Video google: A text retrieval approach to object matching in videos. In *ICCV*, 2003. 2

[38] V. Sydorov, M. Sakurada, and C. H. Lampert. Deep fisher kernels and end to end learning of the Fisher kernel GMM parameters. In *CVPR*, 2014. 2

[39] C. Szegedy, S. Ioffe, and V. Vanhoucke. Inception-v4, inception-resnet and the impact of residual connections on learning. *arXiv:1602.07261v1*, 2016. 1

[40] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. Learning spatiotemporal features with 3d convolutional networks. In *ICCV*, 2015. 1, 2

[41] G. Varol, I. Laptev, and C. Schmid. Long-term Temporal Convolutions for Action Recognition. *PAMI*, 2017. 2

[42] H. Wang and C. Schmid. Action Recognition with Improved Trajectories. In *ICCV*, 2013. 1, 2

[43] L. Wang, Y. Qiao, and X. Tang. Action recognition with trajectory-pooled deep-convolutional descriptors. In *CVPR*, pages 4305–4314, 2015. 2

[44] L. Wang, Y. Xiong, Y. Qiao, D. Lin, X. Tang, and L. Van Gool. Temporal segment networks: Towards good practices for deep action recognition. In *ECCV*, 2016. 2

[45] Z. Xu, Y. Yang, and A. G. Hauptmann. A Discriminative CNN Video Representation for Event Detection. In *CVPR*, 2015. 2

[46] J. Yue-Hei Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici. Beyond short snippets: Deep networks for video classification. In *CVPR*, 2015. 2